PATENT APPLICATION

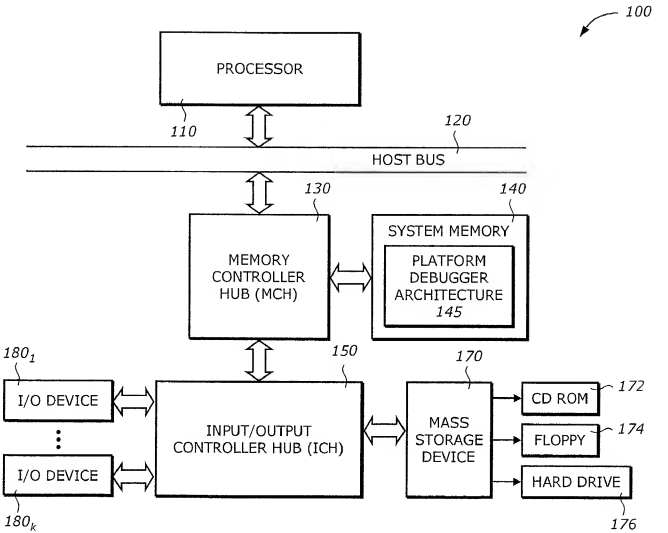

FIG. 1

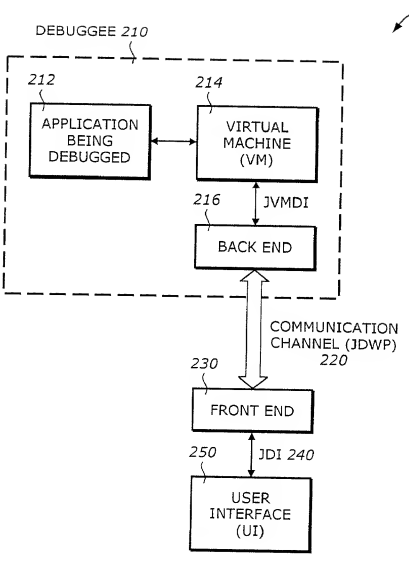
145
DEBUGGEE 210
212
APPLICATION BEING DEBUGGED
214
VIRTUAL MACHINE (VM)
216
JVMDI
BACK END
COMMUNICATION CHANNEL (JDWP) 220
230
FRONT END
250
JDI 240
USER INTERFACE (UI)

FIG. 2

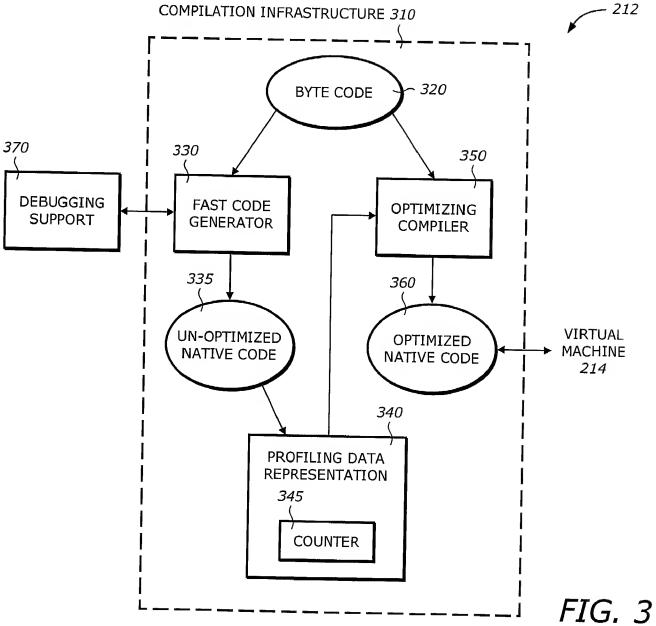
COMPILATION INFRASTRUCTURE 310
212
BYTE CODE
320
370
330
350
DEBUGGING SUPPORT
FAST CODE GENERATOR
OPTIMIZING COMPILER
335
360
UN-OPTIMIZED NATIVE CODE
OPTIMIZED NATIVE CODE
VIRTUAL MACHINE 214
340
PROFILING DATA REPRESENTATION
345
COUNTER

FIG. 3

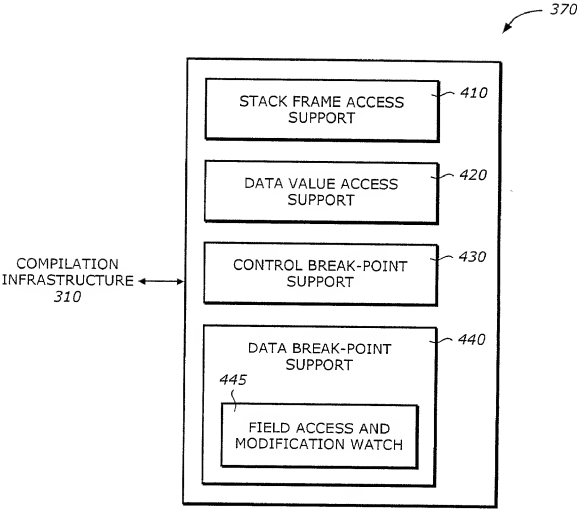
370
STACK FRAME ACCESS SUPPORT
410
DATA VALUE ACCESS SUPPORT
420
CONTROL BREAK-POINT SUPPORT
430
DATA BREAK-POINT SUPPORT
440
445
FIELD ACCESS AND MODIFICATION WATCH
COMPILATION INFRASTRUCTURE
310

FIG. 4

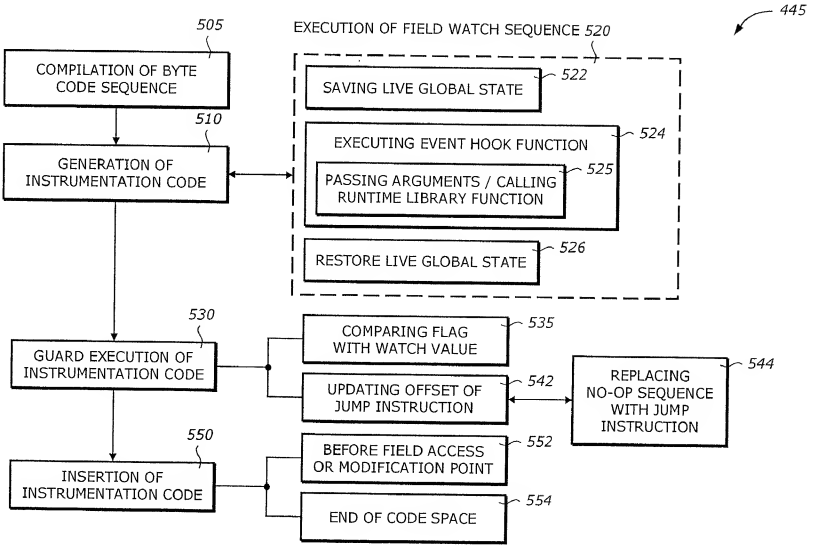
445
505
COMPILATION OF BYTE CODE SEQUENCE
510
GENERATION OF INSTRUMENTATION CODE
530
GUARD EXECUTION OF INSTRUMENTATION CODE
550
INSERTION OF INSTRUMENTATION CODE
EXECUTION OF FIELD WATCH SEQUENCE 520
522
SAVING LIVE GLOBAL STATE
524
EXECUTING EVENT HOOK FUNCTION
525
PASSING ARGUMENTS / CALLING RUNTIME LIBRARY FUNCTION
526
RESTORE LIVE GLOBAL STATE
535
COMPARING FLAG WITH WATCH VALUE
542
UPDATING OFFSET OF JUMP INSTRUCTION
544
REPLACING NO-OP SEQUENCE WITH JUMP INSTRUCTION
552
BEFORE FIELD ACCESS OR MODIFICATION POINT
554
END OF CODE SPACE

FIG. 5

Patent Application

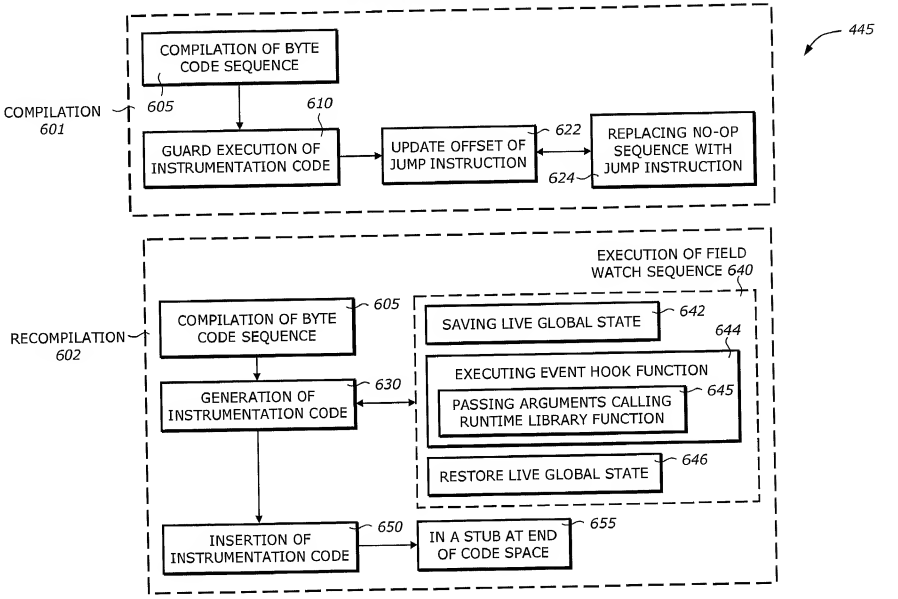

FIG. 6

```
1.  aload 6
2.  getfield #42
3.          mov     ecx, DWORD PTR [esp + 10h]
4.          mov     edx, DWORD PTR [ecx + 04h]
5.          Mov     DWORD PTR [esp + 08h], edx

6.          cmp     BYTE PTR [F10B78h], 01h
7.          jnz     NoWatch
8.          push    0f10b50h
9.          push    ecx
10.         call    10A837A0h

11. iload 4
12. if_cmpne 80

13.    NoWatch:
14.         mov     eax, DWORD PTR [esp + 18h]
15.         cmp     eax, DWORD PTR [ecx + 08h]
16.         jnz     ___
```

FIG. 7

```
1.  dconst_1
2.  aload_0
3.  getfield #159
4.          mov     edx, DWORD PTR [ebp + 08h]

5.          jmp     Watch

6.  i2d

    WatchRet:
7.          fild    DWORD PTR [eax + 44h]

8.  ldc2_w #191
9.  ddiv
10.         fld     QWORD PTR [F7DB08h]
11.         fdivr   st(0), st(1)
12.         fstp    st(1)
    ____

13. areturn
            ____

    Watch:
14.         push    eax

15.         mov     ecx, DWORD PTR [eax + 44h]
16.         mov     DWORD PTR [ebp + FFFFFFB4h], ecx
17.         mov     edx, DWORD PTR [57F710h]
18.         mov     DWORD PTR [ebp + FFFFFFB8h], edx
19.         fstp    QWORD PTR [ebp + FFFFFFC0h]

20.         push    013282a0h
21.         push    eax
22.         call    10A837A0h

23.         fld     QWORD PTR [ebp + FFFFFFC0h]

24.         pop     eax
25.         jmp     WatchRet
```

FIG. 8

```
1.  dconst_1
2.  aload_0
3.  getfield #159
4.          mov     edx, DWORD PTR [ebp + 08h]

5.          jmp     Watch

6.  i2d

    WatchRet:
7.          fild    DWORD PTR [eax + 44h]

8.  ldc2_w #191
9.  ddiv
10.         fld     QWORD PTR [F7DB08h]
11.         fdivr   st(0), st(1)
12.         fstp    st(1)
```

stub

```
Watch:
        push    eax
        mov     ecx, DWORD PTR [eax + 44h]
        mov     DWORD PTR [ebp + FFFFFFB4h], ecx
        mov     edx, DWORD PTR [57F710h]
        mov     DWORD PTR [ebp + FFFFFFB8h], edx
        fstp    QWORD PTR [ebp + FFFFFFC0h]
        push    013282a0h
        push    eax
        call    10A837A0h
        fld     QWORD PTR [ebp + FFFFFFC0h]
        pop     eax
        jmp     WatchRet
```

FIG. 9